Displaying Data Sets with Data Grids

Table of contents

1. Database-Driven Web Applications

A database-driven Beehive web application has the the follow components:

- one or more page flows (= a controller file + JSP pages)
- one or more database control files
- one or more databases

A basic schema for a database-driven application is shown below.

pageflow_database

The Database

The role of the database is as a storehouse for the data. The database persists the data from session to session and is the common object that users view and operate upon.

The Database Control

The database control handles the data traffic with the database. The database control is a Java class that handles the database connection and handles the individual operations on the database. Typically, the methods in the class have associated SQL statements: when a method is called, the associated SQL statement is sent to the database. Any data returned by the database is transformed by the method into an appropriate Java object so it is available to the rest of the web application.

The Page Flow

The Page Flow(s) form the front-end user interface of the application. Through the JSP pages, users can interact with the data in the database, by adding, deleting, and updating the data.

1.1. Rendering Complex Data Sets as HTML

Beehive provides a specialized tag library for viewing and managing the complex data sets associated with database applications: `<netui-data:xxx>` (../apidocs/taglib/taglib-overview-summary.html#netui-data) . The central HTML-rendering tags are

- <netui-data:dataGrid> (../apidocs/taglib/beehive.apache.org/netui/tags-databinding-1.0/dataGrid.html)
- <netui-data:repeater (../apidocs/taglib/beehive.apache.org/netui/tags-databinding-1.0/repeater.html)

The <netui-data:dataGrid> tag is specifically for rendering HTML tables.

The <netui-data:repeater> tag is for more general HTML rendering, including HTML tables, lists, and other shapes.

These tags are capable of rendering the following Java objects into HTML: ArrayList, StringArray, RowSet, ResultSet, Vector, XMLBean, and others.

The tags work by iterating over of the data set, and rendering HTML for each iteration.

To render data, you pass the data set into the tag's `dataSource` attribute.

```
<netui-data:dataGrid name="myGrid"
dataSource="someDatabindingContext.someDataSet">
    ...
</netui-data:dataGrid>
```

The tag then analyzes and iterates over the data set, rendering HTML as it goes.

The remainder of this topic concentrates on using the <netui-data:xxx> tab library to handle data sets.

2. Rendering Data Sets as HTML Tables

2.1. What Objects Can Be Rendered?

The data grid can render any of the following data structures as an HTML table. (The basic rule of thumb is: If you can iterate over the structure, the data grid can render it.)

- any subinterface or implementation of java.util.Collection, including List, Set, SortedSet, ArrayList, HashSet, Vector, etc.
- java.util.ResultSet
- java.util.RowSet
- Object[]
- any XMLBean structure

2.2. How are Objects Rendered?

The data sets above are rendered as multiple rows in an HTML table; a single object in the set is rendered as a single row.

The properties of an object can be rendered as the cells of a single row.

The HTML table has the following basic parts:

Region Name	JSP tag	Rendered HTML Element	Renders	Required
Caption	<netui-data:caption>	<caption>	exactly once	no

Header	<netui-data:header>	<thead>	exactly once	no
Data	<netui-data:rows>	<tbody>	once per row of data in the page	no
Footer	<netui-data:footer>	<tfoot>	exactly once	no

The main body of the data grid (the section that renders inside the <netui-data:rows> tags) can contain the following different kinds of columns.

JSP Tag	Rendered HTML Cell type	Rendered HTML Content Type
<netui-data:headerCell>	<th>	
<netui-data:anchorCell>	<td>	<a>
<netui-data:imageAnchorCell	<td>	<a><img/>
<netui-data:imageCell>	<td>	<img/>
<netui-data:spanCell>	<td>	
<netui-data:templateCell>	<td>	None. Template cell contents are specified by the developer

2.3. CSS Styles for Data Grids

The data grid supports two modes for rendering CSS styles: "default" and "empty".

Under the default mode, a CSS style prefix is prepended to a standard set of CSS style names (see table below). There are different CSS style names for different parts of the rendered HTML. The CSS style used is controlled by writing different prepending prefixes into the HTML.

The prefix is controlled by setting the `styleClassPrefix` attribute of the <netui-data:dataGrid> tag.

```
<netui-data:dataGrid styleClassPrefix="myStyle"/>
```

If no prefix is specified, the default prefix is "`datagrid`".

Under the empty mode, no prefixes are written into the HTML.

The style mode used can be switched by setting the `styleClassPolicy` attribute on the <netui-data:dataGrid> tag.

```
<netui-data:dataGrid styleClassPolicy="empty"/>
```

The following table shows the standard CSS style names written into the HTML under both

the default and standard modes, where <prefix> is the value of the `styleClassPrefix` attribute.

Grid Component	Default Style Name	Empty Style Name
Caption	<prefix>	""
Header Row	<prefix>-header	header
Header Cells	<prefix>	""
Header Cell Sortable	<prefix>-sortable	sortable
Header Cell Sorted	<prefix>-sorted	sorted
Data Row	<prefix>-even	even
Data Row Alternating	<prefix>-odd	odd
Data Cell	<prefix>	""
Data Cell Sorted	<prefix>-sorted	sorted
Footer Row	<prefix>-footer	footer
Table	<prefix>	""

When writing a CSS file for the data grid, the style names should appear as follows:

```
tr.datagrid-header{ ... }
```

Of, if `datagrid` is replaced with `<prefix>`, then:

```
tr.<prefix>-header{ ... }
```

2.4. Data Grid Properties File

The data grid tags are configurable through a `.properties` file. The `.properties` file is referenced by the "`resourceBundlePath`" attribute on the `<netui-data:dataGrid>` tag.

The `.properties` file is compiled (into a MessageResources object) and is referenced by its fully qualified class name.

That is, if the file is located at:

```
WEB-INF/src/datagrid/grid.properties
```

Then the `<netui-data:dataGrid>` tag references it as follows:

```
<netui-data:dataGrid resourceBundlePath="datagrid.grid">
```

The referenced `.properties` file is used to obtain internationalized strings, image names,

and patterns that are used during data grid rendering.

There are a set of default property values specified in the `data-grid-default.properties` file in the `beehive-netui-databinding.jar` file.

The contents of the `data-grid-default.properties` file appears below.

```
datagrid.resource.path=resources/images
datagrid.msg.nodata=No data to display

sort.asc.img=/sortdown.gif
sort.desc.img=/sortup.gif
sort.none.img=/sortable.gif

pager.msg.first=First
pager.msg.previous=Previous
pager.msg.next=Next
pager.msg.last=Last
pager.fmt.banner=Page {0} of {1}
```

These values can be overridden in one of two ways:

- by setting the `resourceBundlePath` on the <netui-data:dataGrid> tag
  ```
  <netui-data:dataGrid resourceBundlePath="datagrid.grid"/>
  ```
- by using the `<netui:attribute>` tag with the `resource` facet to override the properties defined in either the default resource bundle or the user-provided `.properties` file using the `resourceBundlePath`.

Developer specified attributes provided by an `<netui:attribute>` tag will override the values set in both the default and user-defined `.properties` files. For example, to override the text used for the "First" page in a pager, use:

```
<netui:attribute facet="resource" name="pager.msg.first" value="My First
Page"/>
```

3. Setting Up a Basic Grid

This section takes you through a step by step process of setting up a basic grid for 'tabular' data, data such as a ResultSet, RowSet, or any kind of data that can easily be set out like table with rows and columns.

3.1. Setup the Page Flow Web App Template

To set up a grid you first need a source of some tabular data. Below we use an in-memory ArrayList as the source of tabular data. In a real web application, this data source would come from a database, but the in-memory source is sufficient for learning how to render a data grid.

In these setup instructions, we assume that you are using Tomcat as your web application container.

Begin by copying the folder BEEHIVE_HOME/samples/netui-blank into CATALINA_HOME/webapps.

Rename the folder **netui-blank** to the name **gridSample**.

3.2. Create an In-Memory Data Source

Now that the Page Flow template is set up, we are ready to create a data object (i.e., an ArrayList) suitable for rendering as a data grid.

Edit the Controller class

Edit the file `gridSample/Controller.java` as follows. Code to edit is shown in bold.

`gridSample/Controller.java`

```
import javax.servlet.http.HttpSession;

import org.apache.beehive.netui.pageflow.Forward;
import org.apache.beehive.netui.pageflow.PageFlowController;
import org.apache.beehive.netui.pageflow.annotations.Jpf;

@Jpf.Controller(
    simpleActions={
        @Jpf.SimpleAction(name="begin", path="index.jsp")
    },
    sharedFlowRefs={
        @Jpf.SharedFlowRef(name="shared", type=shared.SharedFlow.class)
    }
)
public class Controller
    extends PageFlowController
{
    @Jpf.SharedFlowField(name="shared")
    private shared.SharedFlow sharedFlow;

    public java.util.ArrayList pets = new ArrayList();

    /**
     * Callback that is invoked when this controller instance is created.
     */
    protected void onCreate()
    {
      // build the pets ArrayList

        PetType pet1 = new PetType(1, "American Tabby", "Cat", 20.00);
        PetType pet2 = new PetType(2, "Short Haired", "Cat", 20.00);
        PetType pet3 = new PetType(3, "Long Haired", "Cat", 20.00);
```

```
        PetType pet4 = new PetType(4, "Blue Russian", "Cat", 80.00);
        PetType pet5 = new PetType(5, "Pixy Bob", "Cat", 20.00);
        PetType pet6 = new PetType(6, "Siamese", "Cat", 80.00);
        PetType pet7 = new PetType(7, "Minx", "Cat", 20.00);
        PetType pet8 = new PetType(8, "Bull Mastif", "Dog", 350.00);
        PetType pet9 = new PetType(9, "Dalmatian", "Dog", 20.00);

        pets.add(pet1);
        pets.add(pet2);
        pets.add(pet3);
        pets.add(pet4);
        pets.add(pet5);
        pets.add(pet6);
        pets.add(pet7);
        pets.add(pet8);
        pets.add(pet9);
    }

    /**
     * Callback that is invoked when this controller instance is destroyed.
     */
    protected void onDestroy(HttpSession session)
    {
    }

    // PetType Bean
    public class PetType
    {

        private int petID;
        private String name;
        private String description;
        private double price;

        public PetType() {}

        public PetType(int petID, String name, String description, double
price) {
            this.petID= petID;
            this.name = name;
            this.description = description;
            this.price = price;
        }

        public int getPetID() {return petID;}
        public String getName() {return name;}
        public String getDescription() {return description;}
        public double getPrice() {return price;}
    }
}
```

You now have an in-memory data source of tabular data. In the next step you will point a data grid at this data to render it as HTML.

Compile the Page Flow Web App Template

Before proceeding, you should compile the web app.

The compile step is necessary because you have modified the Controller.java file. In the steps below, where you only modify the JSP page `index.jsp`, you will *not* need to recompile the app to view your changes.

To compile, edit the properties (1) `beehive.home` and (2) `contextPath` in the file `gridSample/WEB-INF/src/build.properties`, for example:

```
beehive.home=C:/apache/apache-beehive-1.0
contextPath=gridSample
```

Then enter the following Ant command:

```
ant -f %CATALINA_HOME%\webapps\gridSample\WEB-INF\src\build.xml clean build
```

3.3. Render a Basic Data Grid

Edit `gridSample/index.jsp` so it appears as follows.

`gridSample/index.jsp`

```
<%@ page language="java" contentType="text/html;charset=UTF-8"%>
<%@ taglib uri="http://beehive.apache.org/netui/tags-databinding-1.0"
prefix="netui-data"%>
<%@ taglib uri="http://beehive.apache.org/netui/tags-html-1.0"
prefix="netui"%>
<%@ taglib uri="http://beehive.apache.org/netui/tags-template-1.0"
prefix="netui-template"%>
<netui:html>
  <head>
    <title>Data Grid</title>
    <netui:base/>
  </head>
  <netui:body>
    <netui-data:dataGrid dataSource="pageFlow.pets" name="petGrid">
        <netui-data:rows>
            <netui-data:spanCell value="${container.item.petID}" />
            <netui-data:spanCell value="${container.item.name}" />
            <netui-data:spanCell value="${container.item.description}"/>
            <netui-data:spanCell value="${container.item.price}"/>
        </netui-data:rows>
    </netui-data:dataGrid>
  </netui:body>
</netui:html>
```

When you visit the URL

```
http://localhost:8080/gridSample/
```

you should see the following page

categoryGrid1

The tabular data is rendered as an HTML table, by the remaining tags (<netui-data:rows> and <netui-data:spanCell).

Here is the renderd HTML table.

```
    Page 1 of 1
<table class="datagrid">
<tr class="datagrid-even">
            <td class="datagrid"><span>1</span></td>
            <td class="datagrid"><span>American Tabby</span></td>
            <td class="datagrid"><span>Cat</span></td>
            <td class="datagrid"><span>20.0</span></td>
</tr>
<tr class="datagrid-odd">
            <td class="datagrid"><span>2</span></td>
            <td class="datagrid"><span>Short Haired</span></td>
            <td class="datagrid"><span>Cat</span></td>
            <td class="datagrid"><span>20.0</span></td>
</tr>
<tr class="datagrid-even">
            <td class="datagrid"><span>3</span></td>
            <td class="datagrid"><span>Long Haired</span></td>
            <td class="datagrid"><span>Cat</span></td>
            <td class="datagrid"><span>20.0</span></td>
</tr>
<tr class="datagrid-odd">
```

```
            <td class="datagrid"><span>4</span></td>
            <td class="datagrid"><span>Blue Russian</span></td>
            <td class="datagrid"><span>Cat</span></td>
            <td class="datagrid"><span>80.0</span></td>
</tr>
<tr class="datagrid-even">
            <td class="datagrid"><span>5</span></td>
            <td class="datagrid"><span>Pixy Bob</span></td>
            <td class="datagrid"><span>Cat</span></td>
            <td class="datagrid"><span>20.0</span></td>
</tr>
<tr class="datagrid-odd">
            <td class="datagrid"><span>6</span></td>
            <td class="datagrid"><span>Siamese</span></td>
            <td class="datagrid"><span>Cat</span></td>
            <td class="datagrid"><span>80.0</span></td>
</tr>
<tr class="datagrid-even">
            <td class="datagrid"><span>7</span></td>
            <td class="datagrid"><span>Minx</span></td>
            <td class="datagrid"><span>Cat</span></td>
            <td class="datagrid"><span>20.0</span></td>
</tr>
<tr class="datagrid-odd">
            <td class="datagrid"><span>8</span></td>
            <td class="datagrid"><span>Bull Mastif</span></td>
            <td class="datagrid"><span>Dog</span></td>
            <td class="datagrid"><span>350.0</span></td>
</tr>
<tr class="datagrid-even">
            <td class="datagrid"><span>9</span></td>
            <td class="datagrid"><span>Dalmatian</span></td>
            <td class="datagrid"><span>Dog</span></td>
            <td class="datagrid"><span>20.0</span></td>
</tr>
    </table>
```

The stages of the rendering are described below.

First the data set is passed to the data grid tag through the `dataSource` attribute:

```
<netui-data:dataGrid name="productsGrid" dataSource="pageFlow.pets">
```

The data grid tag then analyzes the data set and iterates over its parts. In this case, the data grid iterates over the elements in the ArrayList. Typically, a single element in the iteration is rendered as a single row in the HTML table.

During the iteration, that data becomes accessible through the databinding contexts **`container`**, **`item`**, and **`index`**.

- **`container`** refers to the data set passed into the data grid.
- **`container.item`** refers to the current row of the data set. `container.item` refers to each row as the data grid iterates through the set.

- **`container.item.[field_name]`** refers to an individual field of the current row.
- **`container.index`** refers to the integer index of the current row.

The <netui-data:rows> tag controls the rendering of the <tr> tags. Through the <netui-data:rows> tag, you can control the attributes of the rendered <tr> tags, attribute such as `align`, `valign`, and `style`.

In a similar way, the <netui-data:spanCell> tag controls the rendering of the <td> tags.

3.4. Adding a Header

To add a table header, use the <netui-data:headerCell> tag.

Edit `index.jsp` so it appears as follows.

```
<%@ page language="java" contentType="text/html;charset=UTF-8"%>
<%@ taglib uri="http://beehive.apache.org/netui/tags-databinding-1.0"
prefix="netui-data"%>
<%@ taglib uri="http://beehive.apache.org/netui/tags-html-1.0"
prefix="netui"%>
<%@ taglib uri="http://beehive.apache.org/netui/tags-template-1.0"
prefix="netui-template"%>
<netui:html>
  <head>
    <title>Grid Sample</title>
    <netui:base/>
  </head>
  <netui:body>
    <netui-data:dataGrid dataSource="pageFlow.pets" name="petGrid">
        <netui-data:header>
            <netui-data:headerCell headerText="Pet ID Number" />
            <netui-data:headerCell headerText="Name"/>
            <netui-data:headerCell headerText="Description"/>
            <netui-data:headerCell headerText="Price"/>
        </netui-data:header>
        <netui-data:rows>
            <netui-data:spanCell value="${container.item.petID}" />
            <netui-data:spanCell value="${container.item.name}" />
            <netui-data:spanCell value="${container.item.description}"/>
            <netui-data:spanCell value="${container.item.price}"/>
        </netui-data:rows>
    </netui-data:dataGrid>
  </netui:body>
</netui:html>
```

Refresh the browser. You should see the following page.

Page 1 of 1

Pet ID Number	Name	Description	Price
1	American Tabby	Cat	20.0
2	Short Haired	Cat	20.0
3	Long Haired	Cat	20.0
4	Blue Russian	Cat	80.0
5	Pixy Bob	Cat	20.0
6	Siamese	Cat	80.0
7	Minx	Cat	20.0
8	Bull Mastif	Dog	350.0
9	Dalmatian	Dog	20.0

categoryGrid2

The <netui-data:headerCell> renders the following HTML `<tr><th>` row.

```
<tr class="datagrid-header">
    <th class="datagrid">Pet ID Number</th>
    <th class="datagrid">Name</th>
    <th class="datagrid">Description</th>
    <th class="datagrid">Price</th>
</tr>
```

3.5. Controlling the CSS Style

Notice the various `class` attributes in the rendered HTML.

- `class="datagrid"`
- `class="datagrid-header"`
- `class="datagrid-even"`
- `class="datagrid-odd"`

You control the data grid's style by changing the *prefix* rendered in the `class` attributes. If no prefix value is explicitly specified, the default prefix value is *datagrid*.

To put this into practice, first create the following CSS file.

gridSample/resources/css/style.css

```
/* Data grid styles */
.gridStyle
{
color: #111111;
font-size: 14px;
text-align: left;
vertical-align: middle;
padding: 5px 5px 5px 5px;
}
.gridStyle-header
{
background-color: #aaddaa;
color: #FFF;
vertical-align: baseline;
line-height: 18px;
}
.gridStyle-even
{
background-color: #FFFFFF;
color: #111111;
font-size: 14px;
text-align: left;
border-color: #999999;
border-style: solid;
border-width: 1px;
padding-left: 12px;
}
.gridStyle-odd
{
background-color: #dddddd;
color: #111111;
font-size: 14px;
text-align: left;
border-color: #999999;
border-style: solid;
border-width: 1px;
padding-left: 12px;
}
```

To reference this CSS file, edit `index.jsp` as shown below.

gridSample/index.jsp

```
<%@ page language="java" contentType="text/html;charset=UTF-8"%>
<%@ taglib uri="http://beehive.apache.org/netui/tags-databinding-1.0"
prefix="netui-data"%>
<%@ taglib uri="http://beehive.apache.org/netui/tags-html-1.0"
prefix="netui"%>
<%@ taglib uri="http://beehive.apache.org/netui/tags-template-1.0"
prefix="netui-template"%>
<link href="<%=request.getContextPath()%>/resources/css/style.css"
type="text/css" rel="stylesheet"/>
<netui:html>
  <head>
    <title>Grid Sample</title>
```

```
    <netui:base/>
  </head>
  <netui:body>
    <netui-data:dataGrid dataSource="pageFlow.pets" name="petGrid"
styleClassPrefix="gridStyle">
        <netui-data:header>
            <netui-data:headerCell headerText="Pet ID Number" />
            <netui-data:headerCell headerText="Name"/>
            <netui-data:headerCell headerText="Description"/>
            <netui-data:headerCell headerText="Price"/>
        </netui-data:header>
        <netui-data:rows>
            <netui-data:spanCell value="${container.item.petID}" />
            <netui-data:spanCell value="${container.item.name}" />
            <netui-data:spanCell value="${container.item.description}"/>
            <netui-data:spanCell value="${container.item.price}"/>
        </netui-data:rows>
    </netui-data:dataGrid>
  </netui:body>
</netui:html>
```

The resulting HTML page is shown below. (You will need to recompile to see these changes in the browser.)

Page 1 of 1

Pet ID Number	Name	Description	Price
1	American Tabby	Cat	20.0
2	Short Haired	Cat	20.0
3	Long Haired	Cat	20.0
4	Blue Russian	Cat	80.0
5	Pixy Bob	Cat	20.0
6	Siamese	Cat	80.0
7	Minx	Cat	20.0
8	Bull Mastif	Dog	350.0
9	Dalmatian	Dog	20.0

categoryGrid3

3.6. Configuring the Pager

The "pager" controls how many rows are included on a single page of data.

The pager can be configured to appear above and/or below the data grid.

To add a pager, edit `index.jsp` so it appears as follows.

`gridSample/index.jsp`

```
<%@ page language="java" contentType="text/html;charset=UTF-8"%>
<%@ taglib uri="http://beehive.apache.org/netui/tags-databinding-1.0"
prefix="netui-data"%>
<%@ taglib uri="http://beehive.apache.org/netui/tags-html-1.0"
prefix="netui"%>
<%@ taglib uri="http://beehive.apache.org/netui/tags-template-1.0"
prefix="netui-template"%>
<link href="<%=request.getContextPath()%>/resources/css/style.css"
```

```
type="text/css" rel="stylesheet"/>
<netui:html>
  <head>
    <title>Grid Sample</title>
    <netui:base/>
  </head>
  <netui:body>
    <netui-data:dataGrid dataSource="pageFlow.pets" name="petGrid"
styleClassPrefix="gridStyle">
        <netui-data:configurePager pageSize="3" pagerFormat="prevNext"
pageAction="begin.do"/>
        <netui-data:header>
            <netui-data:headerCell headerText="Pet ID Number" />
            <netui-data:headerCell headerText="Name"/>
            <netui-data:headerCell headerText="Description"/>
            <netui-data:headerCell headerText="Price"/>
        </netui-data:header>
        <netui-data:rows>
            <netui-data:spanCell value="${container.item.petID}" />
            <netui-data:spanCell value="${container.item.name}" />
            <netui-data:spanCell value="${container.item.description}"/>
            <netui-data:spanCell value="${container.item.price}"/>
        </netui-data:rows>
    </netui-data:dataGrid>
  </netui:body>
</netui:html>
```

Refresh the browser. You will see the following page.

Page 1 of 3 Previous Next

Pet ID Number	Name	Description	Price
1	American Tabby	Cat	20.0
2	Short Haired	Cat	20.0
3	Long Haired	Cat	20.0

categoryGrid4

3.7. Filtering a Data Grid

To be completed

3.8. Sorting a Data Grid

To be completed

3.9. Nesting Data Grids

To be completed